A Rule-Based Guidance (RBG) System with Graphical Representation of Uncertainty

Stuart Leader
Premier Farnell PLC,
150 Armley Rd, Leeds LS12 2QQ,
Leeds, UK

Ah-Lian Kor
School of Computing, Creative Technologies, and Engineering,
Leeds Beckett University,
Leeds, LS6 3QS, UK
A.Kor@leedsbeckett.ac.uk

Colin Pattinson
School of Computing, Creative Technologies, and Engineering,
Leeds Beckett University,
Leeds, LS6 3QS, UK

Nick Cope
School of Computing, Creative Technologies, and Engineering,
Leeds Beckett University,

***Abstract*— Automated expert systems provide decision support for certain subject domain by prompting users for answers. The answers could either be a 'yes', 'no', or a selected response from multiple choice items. To date, there is no account of any expert system which allows a user to defer or revise his/her response to a particular question. Our Rules-Based Guidance (RBG) System provides such a flexibility to the user. At the same time, its recommendation (or conclusion) is updated accordingly to the revised or new input by user. Legislation compliance is complex and thus rules represented in the knowledge are complex boolean expressions. Easily comprehensible and intuitive information visualisation is necessary to help a lay user understand the reasoning process, derivation of the final recommendation, and also to revisit the state of his/her responses to a fixed set of questions. Thus, RBG System has provided a functionality for transforming boolean expressions (with user associated inputs) into a Directed Acyclic Graph (DAG).**

Keywords—rule-based, decision tree, directed acyclic graph, boolean expression, expert system, legislation compliance

I. INTRODUCTION

In recent years, automated expert systems have become available to users for providing advice and guidance to users in certain specific domains, they usually achieve this by prompting users to answer a set of questions. These systems are then able to provide a recommendation or conclusion to the user based on a set of rules which are deployed to reason with the user-provided answers and knowledge within the knowledge base. Practical examples of such systems are: provide advice and guidance for finding the best car insurance or decision support for curing minor health ailments. However, existing systems do not allow users to defer or backtrack to change their responses. To be truly useful, such a system should allow users the flexibility to modify or defer their responses at any point during the question and answering dialogue session between the user and system. One challenge hereby, is to develop a system in such a way that partial evaluation of the input can be done, based on the questions (note: not all questions) that have already been answered by the user. This means that the system would relay to the user the appropriate conclusion/s that can be given based on the set of incomplete answers. This would allow the user to try out different scenarios with a range of different answers to see the effect of these answers on the recommendation of the system.

Another challenge is to automatically revise the catalogue of questions posed by the system to avoid redundancy (i.e. avoid redundant questioning). The standard straight-forward way of implementing the questioning catalogue would be to simply go through a fixed set of questions and collect all responses. However, based on the specific set of rules in which the domain knowledge is embedded, it could be possible that based on the answers of some questions, other related questions may not be necessary anymore. On the other hand, reasoning with answers to certain specific questions might be inadequate and thus, could require answers from additional questions.

In order to address the above challenges, we have developed a rule-based guidance system (RBG System) for the specific knowledge domain of legislation which supports partial evaluation based on a partial set of user responses (note: it is incomplete due to user's deferral choice) and also the flexibility for users to modify their responses at any point during the evaluation process. In particular, the scope of this project is to address these requirements for an application in the business environment. For an easy-to-use interface, we have developed a graphical representation of the assessment of their given answers and also allows them to intuitively understand the reasoning and rationale for the recommendation outcome by the system, based on the user's inputs (responses to the prompted questions). In the final version of this system, the user will have the option to defer, revoke and amend any of their responses. Subsequently, the user will have the opportunity to view a bespoke diagram that represents the user's responses, inference process and inference outcome. This diagram is the result of transforming a boolean algebra into a Directed Acyclic Graph (DAG). The resulting application will be able to support a range of use cases which requires flexibility where users are not sure of some of the responses, and has the potential to be applied in many industries.

In this paper, we will present the framework whereby a problem domain can be encoded into the system and user input is matched against this recursively until the assessment is complete or a user opts to break out of the cycle. Furthermore, we will present the algorithm which possesses an inherent ability to deal with uncertainty, giving greater flexibility to how the user interacts with the tool without strict prescriptions. We also have developed a front-end application to allow user interaction with the tool and to present the user's custom applicability diagram.

II. RELATED WORK

A. *Expert System Applications*

Expert systems have been deployed in many sectors such as manufacturing, transportation, business, medical, education, etc… In this section we shall discuss relevant work in the areas of product design and legislations. Lai (2007), develops a hybridised expert system using case based reasoning (CBR) and genetic algorithm (GA) for product design. His evaluation shows that the hybrid model outperforms other conventional approaches for creative design. Taylor (1990), builds a hybrid Computer Aided Design (CAD) and expert system for industrial engineers. This hybrid system is called ALFIE (Auxiliary Logistics For Industrial Engineers) and it integrates a rule-based with a model-based approach to expert knowledge. Additionally, it is used alongside a conventional CAD system to assist design engineers in the design process. A frame model of an expert system for product design has been developed by Wang and colleagues (2007). The three modules in the expert system are: product sepcifications, design support, and product plan evaluation. Zarandi and colleagues (2011) have integrated the material selection approach for sustainable products and an expert system (containing eco-design expert knowledge) to effect sustainable product design. Results of their use cases indicate of an increased possibility of manufacturing more sustainable products; increased time as well as cost savings in design and production.

Olugu et. al (2012) have developed a fuzzy rule-based expert system to support a successful closed-loop supply chain within the automotive industry. Expert systems have also been deployed to effectively support decision making in environmental management, for example, improving water quality within the city (Cheng, et. al, 2003). Knowledge bases have been created to give advice for legislative history research (Hardy, 1993), provide guidance on public adminstration related legislations (Johnson and Mead, 1991), and legislative drafting (Cammelli and Mead, 1990). REPIC provides guidance on WEEE, Waste batteries, Accumulators and Packaging compliance as well as regulations to UK companies. As a result of this, its users (e.g. local authorities, waste companies, retailers, charities, re-use organisations, treatment companies, etc…).

B. *Transformation of Boolean Expressions to Directed Acyclic Graphs*

According to Watcher and Haenni (2006), Propositional Directed Acyclic Graph (PDAG) is a new graph-based language (or data structure) that is used to represent a Boolean function. A PDAG has the following form: (i) leaves that are labelled with true, false or a boolean variable; non-leaves are logical and (this node will have at least one child), or (this node will have at least one child), not (this node has exactly one child). However, such representation is not intuitive and will not be easily comprehensible for lay users. Thus, we have employed the use of a Directed Acyclic Graph (DAG) to represent more than one rules (in the form of boolean algebras) in the knowledge base. An example of a DAG generated by our RBG system is depicted in Figure 1. It consists of the following elements: (i) nodes to represent propositions (e.g. A, B, C, D, E) and the truth value will be determined; (ii) directed edges which are represented by "arrows" from one node to another. The relationship represented in between two adjacent nodes

Iin Fgure 1) is an ‘and’ relationship; (iii) a root node which has no parents (i.e. the “start” node in Figure 1); (iv) leaf nodes which are nodes without any children (i.e. the “end” node in Figure 1) and its possible value could be “APPLICABLE” (aka true), “NOT_APPLICABLE” (aka false), or “POTENTIALLY_APPLICABLE” (aka undecidable).

III. SYSTEM LIFECYCLE

A. *Requirements Phase*

Expert systems provide a mechanism to assist individuals in making otherwise convoluted decisions over a given problem domain. A user’s applicability to a set of rules in the knowledge base can be determined through an interview process, where key information is elicited from the user through a series of questions. An environment in RBGS consists of a collection of the user’s responses that describe their idiosyncratic situation. As the user progresses through a set of questions for a specific problem, the environment gradually builds up. After the user has responded to each question, the environment grows and the system inference engine will update its conclusion accordingly based on reasoning with the user given responses and also the appropriate rules in the knowledge base. In an ideal scenario, the user possesses adequate level knowledge necessary for answering all the questions comprehensively. However, the proposed system will could cope with uncertainty where some of the questions are not answered due to the following possible reasons: user does not know the answer; user deliberately chooses not to answer at that point in time. This is achievable by means of allowing the deferral of responding to questions. A given set of user responses will be evaluated against a given ruleset in the knowledge base. The conclusion that is drawn for the applicability of a legislation based on user inputs could be: APPLICABLE, NOT_APPLICABLE, or POTENTIALLY_APPLICABLE (neither yes nor no). An extreme scenario will occur when every question is deferred, and thus every rule will be described as POTENTIALLY_APPLICABLE. RBGS will also seek to bestow the user with justification of the applicability assessments it has made. Each rule can therefore be described as a Directed Acyclic Graph (DAG), colour-coded to facilitate the interpretation of its applicability.

For each rule encoded in the system have multiple paths to applicability. For example, we have the following propositions in the knowledge base: A, B, C, D, and E. In order to fire a rule, R_1, the following boolean algebra will have to be true:

$$R_1 = A \wedge \big(B \wedge (C \vee D) \vee (C \wedge E)\big) \qquad (1)$$

Expand equation (1) and we have:

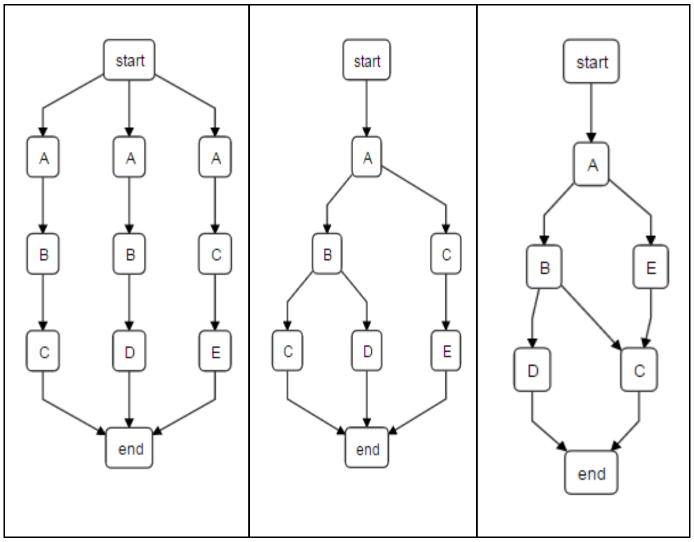
$$(A \wedge B \wedge C) \vee \ (A \wedge B \wedge C) \vee \ (A \wedge B \wedge C) \qquad (2)$$

Thus, the rule will be deemed applicable if the following conditions are satisfied: $\{A \wedge B \wedge C\}$ or $\{A \wedge B \wedge D\}$ or $\{A \wedge C \wedge E\}$. However, the boolean algebra expression will not be easily comprehensible to a lay end-user. To facilitate easy understanding, an algorithm will be required to transform the boolean algebra in equation (2) into a minimal DAG, where minimal is defined as having the fewest number of nodes and edges by consolidating common nodes across the possible applicability sets as previously defined. Figure 1 depicts three correctly generated DAGs. However, the first graph is a *maximal* one where all the possible clauses in equation (2) have been represented. The second graph reduces the number of nodes and paths in the graph by merging the relevant ones. However, only some of the relevant nodes and paths have been merged and thus, the resultant graph is a *partial consolidated* graph. Graph 3 represents a *minimal* graph where all the relevant nodes and paths have been appropriately merged and at the same time maintain the integrity of the reasoning mechanism. Particular consideration must be given to the graphing algorithm so as to ensure that such integrity is maintained. In certain scenarios, nodes representing individual conditions must be duplicated and a separate path through the DAG created in order to avoid creating invalid paths as a result of the following actions: by passing compulsory nodes; introducing edges that act as shortcuts; introducing additional nodes that have impact on the minimum sets of nodes but must be satisfied for the rule to be applicable.

A suitable application of this work could include medical diagnosis, helping to provide guidance on ailments based on observed symptoms. The software may be used in the insurance industry to decide whether to offer services to certain individuals. As a further example, the tool could be used to support support designers in the selection of mandatory and optional specifications associated with a particular product.

Maximal	**Partial Consolidation**	**Minimal**

This project is funded by InnovateUK under the KTP scheme.



Key: start – session starts; end – session ends where a conculsion will be drawn based on inputs for the relevant propositions (from A to E).

Figure 1: Directed Acyclic Graph (DAG)

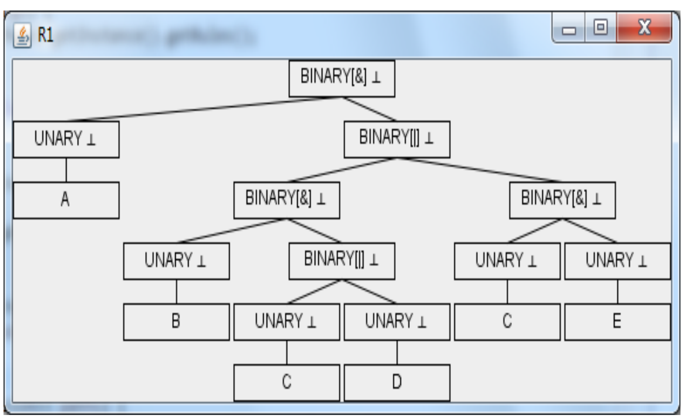
IV. RBG SYSTEM: KNOWLEDGE REPRESENTATION AND REASONING

A. Knowledge Representation and Reasoning

Rules are stored as binary AND-OR tree representations in the knowledge base. Parent nodes can be an AND (represented by &) or OR (represented by |) node and can have the state DETERMINED (note: if user has provided a response to the corresponding question) or UNDETERMINED (note: if user has NOT provided a response to the corresponding question). Figure 2 shows that the tree supports both unary and binary expressions and it is a representation of R_1 in equation 1.

Figure 2: Binary AND-OR Tree Representation of a rule (example R_1 in equation 1)

Let us replace A to E with real life related propositions in the medical field. For the FLU example, the representation is as follows - A: temp>38°C; B: fever; C: headache; D: fatigue; E: chills. A function has been written to traverse the tree in Figure 2. The outcome of the traversal is a list of possible sets of conditions that must be satisfied in order to fire the rule R_1 (or in other words, APPLICABLE). These sets of conditions are aligned to the expansion of equation (1) as shown in equation (2). However, we work with sets to represent the clauses (in our prototype) for easy implementation. As an example, the clause, A $\wedge$ B $\wedge$ C, is represented as [A, B, C]. Consequently, in our RBG prototype, equation (2) is represented as follows : { [temp>38°C , fever , headache] , [temp>38°C , fever , fatigue] , [temp>38°C , headache , fever] }. If any of the clauses in the set is satisfied then this implies R_1 is applicale. To reiterate, user responses for each proposition in the clause could be APPLICABLE (for a positive response), NOT_APPLICABLE (for a negative response), and POTENTIALLY_APPLICABLE (for a deferred response). However, these clauses cannot be directly parsed by the graphing algorithm because it is not guaranteed that they will produce a *minimal* graph (as depicted in Figure 1).



B. Minimal Graph

In order to produce the minimal graph, the sets of clauses must firstly be passed through a sorting algorithm to order the nodes contained in each set and the sets relative to each other. For example, we have the following two clauses for a particular rule, R: [B, C, A], and [D, A, C, B]. These two clauses will be ordered (in this case, frequency and alphabetical order) and the resultant set is as follows: {[A, B, C], [~~A, B, C, D~~]}. We apply Schopenhauer's Second Form of the Principle of Sufficient Reasoning that is *The Principle of Sufficient Reason of Knowing (principium rationis sufficientis cognoscendi)* which asserts that *if a judgment is to express a piece of knowledge, it must have a sufficient ground or reason, in which case it receives the predicate true* (Cartwright, 2012). Thus, if [A, B, C] is true, then it is sufficient to evaluate R as true without having to evaluate the truth value for D. To draw the minimal graph, we graph, we shall omit D, delete the clause [A, B, C, D] from the resultant set, {[A, B, C], ~~[A, B, C, D]~~}. Consequently, we shall be left with the suffient clause, [A, B, C] in the set, which will be used to draw the minimal graph with associated user response for each proposition and a possible conclusion that could be drawn based on the user responses (i.e. APPLICABLE for true, NOT_APPLICABLE for false, POTENTIALLY_APPLICABLE for an undecidable conclusion due to insufficient reason of knowing). The target graph will represent multiple possible paths to achieve applicaibility of a given goal (i.e. the goal is APPLICABLE). To reiterate, the minimal graph in Figure 1 will have three possible paths {[A, B, C], [A, B, D], [A, C, E]}. Shared conditions will be represented as shared nodes in the graph with the condition that integrity of the reasoning is not violated. For example, for a rule, R, to be true (or in the context of the RBG system, APPLICABLE), the following set of clauses must be true: {[A, B, D, E, I], [A, B, C, E, H]}. A minimal graph is drawn in Figure 3. However, the integrity of will have to maintained.

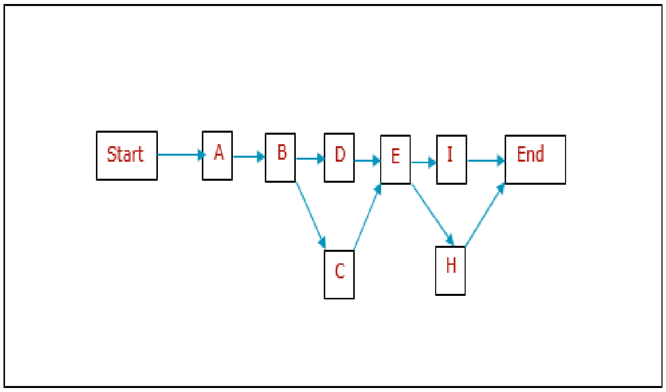


Figure 3: Minimal Graph with integrity violation

In order to evaluate the integrity of the graph, we extract the possible paths from the graph and evaluate them with the original set of clauses. Based on the minimal graph drawn, we have a set of the following possible paths: {[A, B, D, E, I], [A, B, D, E, H], [A, B, C, E, I], [A, B, C, E, H]}. This abstracted set is evaluated against the original set of cluases and we have two additional clauses [A, B, D, E, H] and [A, B, C, E, I]. Thus, this means that the integrity of the graph has been violated. In order to address this problem, we draw a minimal graph without integrity violation as depicted in Figure 4.

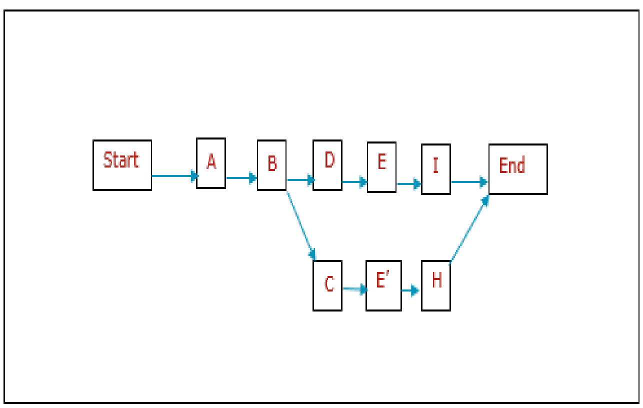


Figure 4: Minimal Graph without integrity violation

In Figure 4, node E cannot be shared because it will affect the integrity of the graph and consequently, we duplicate this node in order to maintain a distinctive flow to the end node. Based on Figure 4, the abstracted possible paths are: {[A, B, D, E, I], [A, B, C, E', H]}. This set of clauses are evaluted against the original set of clauses and they are similar. Thus, we could conclude that the graph is minimal without integrity violation and we have developed an algorithm for transforming boolean algebra expressions to to such form of DAG graphs.

C. RBG System Implementation

We have developed our RBG system using Java and hence, the syntax depicted will be Java syntax. The knowledge base will consist of domain knowledge represented as Boolean algebras using Java-like syntax in text files. As an example, R_1 in equation will be represented as follows:

$$R_1 = A \ \& \ ((B \ \& \ C \mid D) \mid E) \qquad (3)$$

This syntax was chosen to encode the requirements because this form of representation could be easily parsed into DAG graphs using existing drawing tools.

In our RBG system, we could also represent a rule with relation operators (==, <=, >, etc) or negation (!A). An example of such a rule, R_2, is shown in equation (4).

$$R_2 = !A \,\&\, ((B > 12 \,\&\, C \mid D) \mid E == \text{“value”}) \quad (4)$$

In order to build a more complex domain knowledge, the knowledge base could comprise composite rules. As an example, rule R_3 contains rule R_2 in its definition. See equation (5).

$$R_3 = A \mid B \,\&\, R_2 \qquad (5)$$

Rules and facts are parsed by the parser generator, Java Compiler Compiler[1] (JavaCC). Java Tree Builder[2] (JTB) takes a plain JavaCC grammar file as input and automatically generates the following: a set of syntax tree classes based on the productions in the grammar, utilising the Visitor design pattern; proper annotations to build the syntax tree during parsing and a custom internal representation of the inputted rule. This internal representation is a binary AND-OR tree (see Figure 2) and maintains the precedence of operators as defined in the original Boolean expression. User data is evaluated against the binary tree in a recursive fashion, and this traversal algorithm is able to filter the next question to be given to the user accordingly.

V. CONCLUSION

This paper has discussed how a rule-based system could be devloped to handle uncertainty which addresses deferred user inputs (i.e when a response is “POTENTIALLY_APPLICABLE” or in other word, undecidable. RBG is flexible because it allows users to defer and revise the responses followed by a corresponding and automated update of the system’s recommendation (or conclusion) based on the principle of sufficient reason of knowing. A minimal graph has been built to depict the following: as an explanation generator; inference process; recommnedation/conclusion arrived at; status of user’s responses. Rigorous testing has been conducted on the RBG system. Testing on the rule parser has been performed using several checks: if a rule is always true or false, this gets identified. Parser error will automatically halt the system. This means that the system administrator of this expert system needs to ensure that the rule parsing is correct and that the rules have been coded properly. The minimal graph generator has been evaluated. However, the generated diagram is not not necessarily always minimal. Thus, there is a need to refine or revise the minimal graphing algorithm based on fequency of nodes and ordering based on alphabetical order.

REFERENCES

[1] Cammelli, A., and Socci, F. (1990). Lexis: A Legal Expert System for Improving Legislative Drafting. In Tjoa, Min A, et. al. (eds.). Database and Expert Systems Applications, Springer Verlag, pp 405-409.

[2] Cartwright, D. E, et. al (2012). Arthur Schopenhauer: On the Fourfold Root of the Principle of Sufficient Reason: On Vision and Colours: On Will in Nature, Cambridge University Press.

[3] Cheng, H, et. al. (2003). Applications of Artificial Intelligence for Management and Control of Pollution Minimization and Mitigation Processes, Engineering Applications of Artificial Intelligence Volume 16, Issue 2, March 2003, pages 159–166.

[4] Hardy, I. T. (1993). Creating an Expert System for Legislative History Research: Project CLEAR’s “Lexpert”, 85 Law Libr. J. 239.

[5] Johnson, P. and Mead D. (1991). Legislative knowledge base systems for public administration: some practical issues, Proceedings of ICAIL '91 Proceedings of the 3rd international conference on Artificial intelligence and law, pp. 108-117.

[6] Lai, C. (2007). The Expert System of Product Design Based on CBR and GA, Proceddings of International Conference on Computational Intelligence and Security Workshops, 15-19 Dec, 2007, Harbin.

[7] Morgan, J. M., and Liker, J. K. (2006). The Toyota Product Development System, Productivity Press, New York.

[8] Olugu, E. U., et. al (2012). An expert fuzzy rule-based system for closed-loop supply chain performance assessment in the automotive industry, Expert Systems with Applications, Volume 39, Issue 1, January 2012, pages 375–384.

[9] Taylor, N. K. (1990). An Expert System To Assist In Design, an unpublished dissertation, URL: University of Nottingham, URL: http://www.macs.hw.ac.uk/~nkt/acabio/ALFIE.pdf, accessed date: [15th May, 2016].

[10] Wang, C., et. al. (2007). One Frame of Expert System for Product Design, Proceedings of Second International Innovative Computing, Information and Control, 5-7, September, ICICIC '07, Kumamoto.

[11] Watcher, M., and Haenni, R. (2006). Propositional DAGs: a New Graph-Based Language for Representing Boolean Functions, American Association for Artificial Intelligence.

[12] Zarandi, M. H. F., et. al. (2011). A Material Selection Methodology And Expert System For Sustainable Product Design, The International Journal of Advanced Manufacturing Technology, December 2011, Volume 57, Issue 9, pp 885-903

---

[1] https://javacc.java.net/

[2] http://compilers.cs.ucla.edu/jtb/jtb-2003/